skillcrush

SINATRA

cheatsheet

SINATRA

Sinatra

Sinatra is a Ruby web framework.

Many call it a “lightweight web framework” because unlike Rails, Sinatra doesn’t FORCE you to do things a certain way. Instead, it gives you the tools and you can configure it as you see fit.
Sinatra is a great place to start because you can take it in, one bit at a time and get up and running on the web in a matter of MINUTES!

www.sinatrarb.com

HOW TO INSTALL SINATRA

To install Sinatra:

```
gem install sinatra
```

To use Sinatra in a script:

```
require ‘sinatra’
```

skillcrush

SINATRA

cheatsheet

YOUR FIRST SINATRA APP

To create your first Sinatra app, create a new file called sinatra.rb and add the following:

```
require 'sinatra'

get '/' do
     "hello world"
end
```

Start the Sinatra server by running the following command in the terminal:

```
ruby sinatra.rb
```

(change the name of the file to the name of YOUR file)

You will know the server is running when you see something like this:

```
[2015-03-09 20:47:45] INFO  WEBrick 1.3.1
[2015-03-09 20:47:45] INFO  ruby 1.9.3 (2013-11-22) [x86_64-darwin12.5.0]
== Sinatra/1.4.5 has taken the stage on 4567 for development with backup from WEBrick
[2015-03-09 20:47:45] INFO  WEBrick::HTTPServer#start: pid=32959 port=4567
```

Then, in your browser, navigate to http://localhost:4567. You should see:

hello world

To shut down your Sinatra server:

```
^C (control-C)
```

skillcrush

SINATRA

cheatsheet

CREATING & USING VIEWS

View files in Sinatra are called ERB templates.

Each ERB template can have a mix of HTML and ERB tags:

```
<!DOCTYPE html>
<html>
<head>
     <meta charset="utf-8">
     <title>Sinatra App</title>
</head>
     <body>
          <h1><%= @myvariable %></h1>
     </body>
</html>
```

To create views, first create a views folder in the same location as your sinatra.rb file. Inside that views folder, create your templates, starting with index.erb.

There are two types of ERB tags. This type of tag processes the Ruby contained inside the tags:

```
<% %>
```

This type of tag processes the Ruby AND prints the result:

```
<%= %>
```

To reuse the same boilerplate HTML, you can create a layout.erb template. Replace your dynamic page content with:

```
<%= yield %>
```

Like so:

```
<!DOCTYPE html>
<html>
<head>
     <meta charset="utf-8">
     <title>Sinatra App</title>
</head>
     <body>
          <%= yield %>
     </body>
</html>
```

Then, you can specify your templates in your requests like so:

```
get '/' do
    erb :index
end
```

Or:

```
get '/' do
    erb :”index”
end
```

Where :index or :”index” are the name of your template files.

skillcrush

SINATRA

cheatsheet

OTHER TYPES OF HTTP REQUESTS

A GET request gets data and displays it:

```
get '/' do
    # do something
end
```

A POST request creates new data:

```
post '/' do
    # create something
end
```

A PUT request replaces existing data:

```
put '/' do
    # replace something
end
```

A DELETE request deletes data:

```
delete '/' do
    # annihilate something
end
```

skillcrush

SINATRA

cheatsheet

Passing Dynamic URL Parameters

To pass a URL parameter such as an ID to a request, use params:

```
get '/:id'
     id = params[:id]
end
```